**APPLICAZIONI WEB: LINGUAGGI E ARCHITETTURE**

**a.a. 2017/2018**

**Progetto di laboratorio**

Alessandro Salogni, Riccardo Perotti, Simone Coppeta

20013907, 20013787, 20016455 @studenti.uniupo.it

1. **Istruzioni iniziali**

Collegarsi alla repository GitHub: <https://github.com/AlessandroSalogni/layw-appwebvc-2018> e scaricare il progetto. Una volta scaricato, aprirlo con Visual Studio 2017 (v. 15.7.3 o superiore), con SDK .NET core 2.1.

Se ci si vuole loggare nell’applicazione come medico, è sufficiente sfruttare l’autenticazione di Google oppure quella di Facebook. Al primo login da medico, naturalmente, non ci saranno pazienti associati; pertanto ci si deve loggare come amministratore premendo il link apposito (*login as administrator*) ed inserire mail e password (account: [*lookafteryourweight@gmail.com*](mailto:lookafteryourweight@gmail.com) password: *admin*). Dopodiché sarà possibile collegare dei pazienti all’account medico precedentemente creato.

Non è necessario alcuno script SQL o altre impostazioni iniziali. I dati sorgenti sono estratti tramite API REST, eseguite su un server Heroku, ove è contenuto anche il Database.

Per una visione d’insieme dell’architettura totale del progetto *LAYW*, si allega la documentazione relativa anche alla parte di Reti 2.

1. **Scelte implementative e note particolari**

L’applicazione Web dialoga con uno strato di API REST definite sul server Heroku, il quale rappresenta l’accesso al modello dei dati. I dati sono visualizzati nell’applicazione web servendosi del framework Kendo e di alcuni script JS. Le API REST su server Heroku sono state sviluppate in Java per il progetto di Reti 2, e per il momento non prevedono ancora un’autenticazione, sicuramente necessaria in un sistema pubblicato.

Sul server Heroku, oltre alle API REST, è implementato anche un Publisher MQTT. L’applicazione Web pertanto implementa un Subscriber MQTT per gestire le notifiche da visualizzare al medico.

Nella homepage del medico si è deciso di rappresentare i pazienti che hanno superato specifici obiettivi per permettere al dottore di avere una visione generale ma allo stesso tempo efficacie della situazione globale dei suoi pazienti, una volta effettuato il login; le informazioni più specifiche di un paziente (come la dieta, il piano di allenamento, dati sulle sue attività giornaliere), si possono analizzare in dettaglio tramite un apposito menù.

Il sistema è pensato per una piccola utenza per ogni medico. Si suppone che i casi che segue un determinato medico siano sufficientemente pochi per consentirgli un’attenta analisi giornaliera. Si stimano 5-10 pazienti massimo per ogni medico.

Le notifiche sull’applicazione Web arrivano casualmente: per permettere di vederne il funzionamento in qualsiasi momento, è presente un raspberry sempre in funzione che periodicamente è responsabile della pubblicazione dei messaggi MQTT, che verranno notificati. Nella realtà le notifiche arriverebbero solo quando il paziente modifica il suo peso, raggiunge determinati obiettivi, ecc.

I dati sul Database di Heroku potrebbero essere inconsistenti, poiché sono dati inseriti manualmente, per permettere la prova del funzionamento dell’applicazione web. Se si notassero delle incongruenze, è probabile che il problema sia la popolazione di prova fatta a mano.

Le password degli amministratori sono criptate in MD5. Un amministratore può associare pazienti e medici e aggiungere altri amministratori specificando solo una mail valida. Ci si può iscrivere al sistema soltanto come medico: per essere amministratore, si deve per forza essere invitati da altri amministratori (i quali, specificando un’indirizzo mail, inviano in automatico una mail al nuovo admin con le sue credenziali, di avvenuta registrazione). Questo è stato fatto per garantire maggior controllo sul sistema.

Ogni tab di ogni paziente carica i dati al click della tab, in modo da evitare un caricamento di dati totale piuttoto pesante, ed in modo da avere anche le tab sempre aggiornate. Si è scelto di fare tutto in tab e non in cambio pagina poiché si ritiene che sia più dinamico e veloce, piuttosto che spostarsi in una nuova pagina e cambiare view.

Due tipi di login, perché staccati, pro e contro.

***Limiti***

Il piano base dell’hosting della piattaforma su Heroku permette 20 connessioni in parallelo massime al Database. Inoltre, una volta al giorno il programma in deploy su Heroku si riavvia. Questi limiti potrebbero portare al fallimento di richieste REST, o a piccoli malfunzionamenti del programma, ma in linea di massima non dovrebbero presentarsi.

1. **Descrizione del modello dei dati**

Nella repository GitHub: <https://github.com/AlessandroSalogni/layw-appwebvc-2018> è presente il file “Documentazione\_CINI.pdf”, una documentazione del progetto generale dove è presente la descrizione delle API REST sviluppate (in Java poiché previste per il progetto di Reti 2) – in fondo al documento.

Per la gestione degli utenti amministratore è presente invece un DB locale all’applicazione Web contenente una sola tabella, avente per semplicità solo due campi: *e-mail* e *password*.

1. **Descrizione del modello ad oggetti**

Si riportano di seguito i diagrammi UML che rappresentano le classi sviluppate.